# SOFTWARE ENGINEERING

## CO3001

### CHAPTER 4 – REQUIREMENTS ENGINEERING

**WEEK 4, 5**

# TOPICS COVERED

✓ Functional and non-functional requirements

✓ Requirements engineering processes

✓ Requirements elicitation

✓ Requirements specification

✓ Requirements validation

✓ Requirements change

# REQUIREMENTS

# REQUIREMENTS ENGINEERING

✓ The process of establishing the **services** that the customer requires from a system and the **constraints** under which it operates and is developed.

# WHAT IS A REQUIREMENT?

Requirement engineering = establishing the **services** that the customer requires from a system and the **constraints** under which it operates and is developed.

✓ Requirement = the descriptions of
- the system services
- and constraints

✓ It may range
- from a high-level abstract statement
- to a detailed mathematical functional specification.

✓ May serve a dual function
- The basis for a bid for a contract - must be open to interpretation;
- The basis for the contract itself - must be in detail;

# TYPES OF REQUIREMENT

✓ User requirements
- Written for customers.
  - statements in natural language + diagrams of the services the system provides and its operational constraints.

✓ System requirements
- (For) developers/contractor
  - detailed descriptions of the system's functions, services and operational constraints
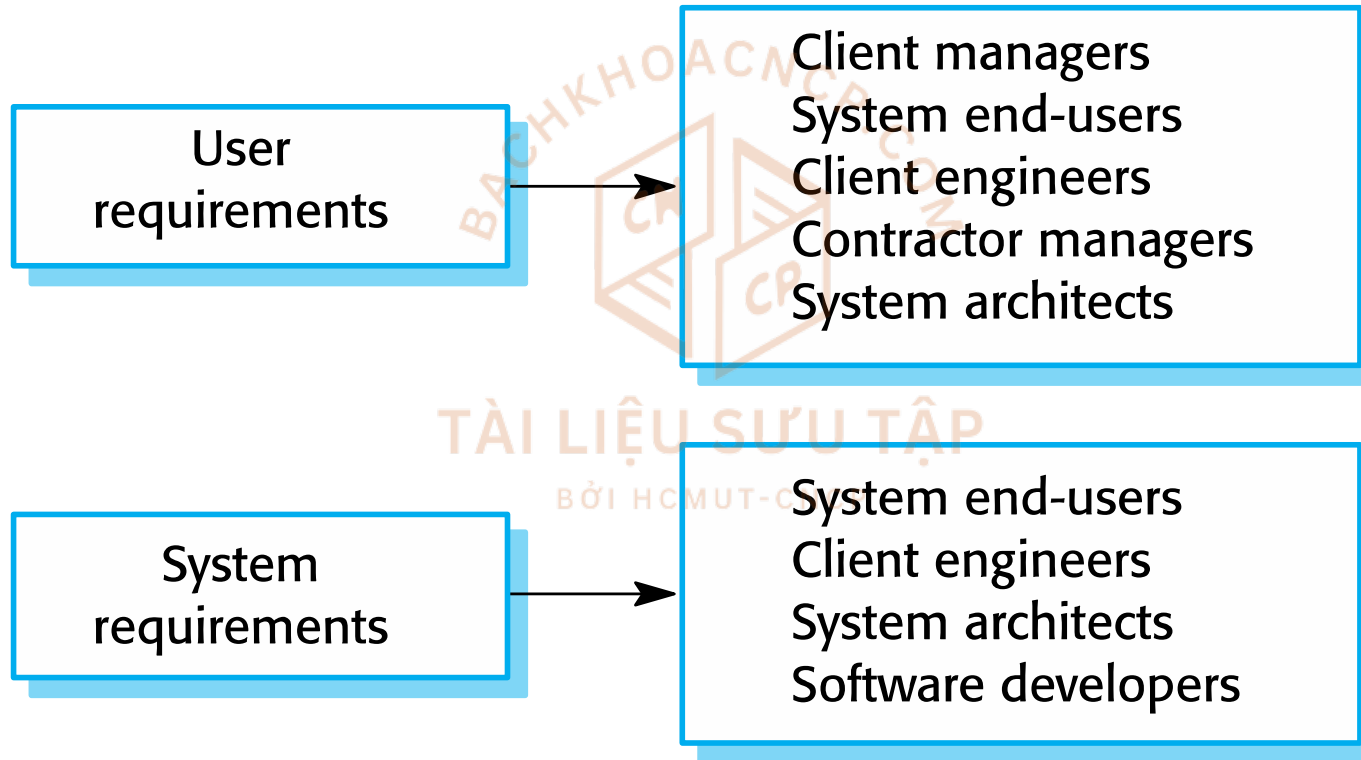
# USER AND SYSTEM REQUIREMENTS EXAMPLE

### User requirements definition

1. The Mentcare system shall generate monthly management reports showing the cost of drugs prescribed by each clinic during that month.

### System requirements specification

**1.1** On the last working day of each month, a summary of the drugs prescribed, their cost and the prescribing clinics shall be generated.

**1.2** The system shall generate the report for printing after 17.30 on the last working day of the month.

**1.3** A report shall be created for each clinic and shall list the individual drug names, the total number of prescriptions, the number of doses prescribed and the total cost of the prescribed drugs.

**1.4** If drugs are available in different dose units (e.g. 10mg, 20mg, etc) separate reports shall be created for each dose unit.

**1.5** Access to drug cost reports shall be restricted to authorized users as listed on a management access control list.

# READERS OF DIFFERENT TYPES OF REQUIREMENTS SPECIFICATION

User requirements → Client managers
System end-users
Client engineers
Contractor managers
System architects

System requirements → System end-users
Client engineers
System architects
Software developers

# SYSTEM STAKEHOLDERS

✓ Any person or organization who is affected by the system in some way and so who has a legitimate interest

✓ Stakeholder types
  ▪ End users
  ▪ System managers
  ▪ System owners
  ▪ External stakeholders

# STAKEHOLDERS IN THE MENTCARE SYSTEM

| Stakeholder | Why? - Role |
|---|---|
| Patients | whose information is recorded in the system |
| Doctors | responsible for assessing and treating patients |
| Nurses | coordinate the consultations with doctors and administer some treatments |
| Medical receptionists | manage patients' appointments |
| IT staff | responsible for installing and maintaining the system |
| Medical ethics manager | ensure that the system meets current ethical guidelines for patient care |
| Health care managers | obtain management information from the system |
| Medical records staff | responsible for ensuring that system information can be maintained and preserved, and that record keeping procedures have been properly implemented. |

# AGILE METHODS AND REQUIREMENTS

✓ Many agile methods argue that producing detailed system requirements is a waste of time as requirements change so quickly.

✓ The requirements document is therefore always out of date.

✓ Agile methods usually use incremental requirements engineering and may express requirements as 'user stories'

✓ This is practical for business systems but problematic for systems that require pre-delivery analysis (e.g. critical systems) or systems developed by several teams.

# FUNCTIONAL AND
# NON-FUNCTIONAL REQUIREMENTS

# FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS

✓ Functional requirements
- Statements of services the system should provide, how the system should react to particular inputs and how the system should behave in particular situations.
- May state what the system should not do.

✓ Non-functional requirements
- Constraints on the services or functions offered by the system such as timing constraints, constraints on the development process, standards, etc.
- Often apply to the system as a whole rather than individual features or services.

- Domain requirements
  - Constraints on the system from the domain of operation

# FUNCTIONAL REQUIREMENTS

✓ Describe functionality or system services.

- Functional user requirements may be high-level statements of what the system should do.
- Functional system requirements should describe the system services in detail.

# MENTCARE SYSTEM: FUNCTIONAL REQUIREMENTS

1. A user shall be able to search the appointments lists for all clinics.

2. The system shall generate each day, for each clinic, a list of patients who are expected to attend appointments that day.

3. Each staff member using the system shall be uniquely identified by his or her 8-digit employee number.

# REQUIREMENTS IMPRECISION

✓ Problems arise when requirements are not precisely stated.
- Ambiguous requirements may be interpreted in different ways by developers and users.

✓ For the term 'search' in requirement 1
- User intention – search for a patient name across all appointments in all clinics;
- Developer interpretation – search for a patient name in an individual clinic. User chooses clinic then search.

# REQUIREMENTS COMPLETENESS AND CONSISTENCY

✓ Requirements should be both complete and consistent.

✓ Complete
- They should include descriptions of all facilities required.

✓ Consistent
- There should be no conflicts or contradictions in the descriptions of the system facilities.
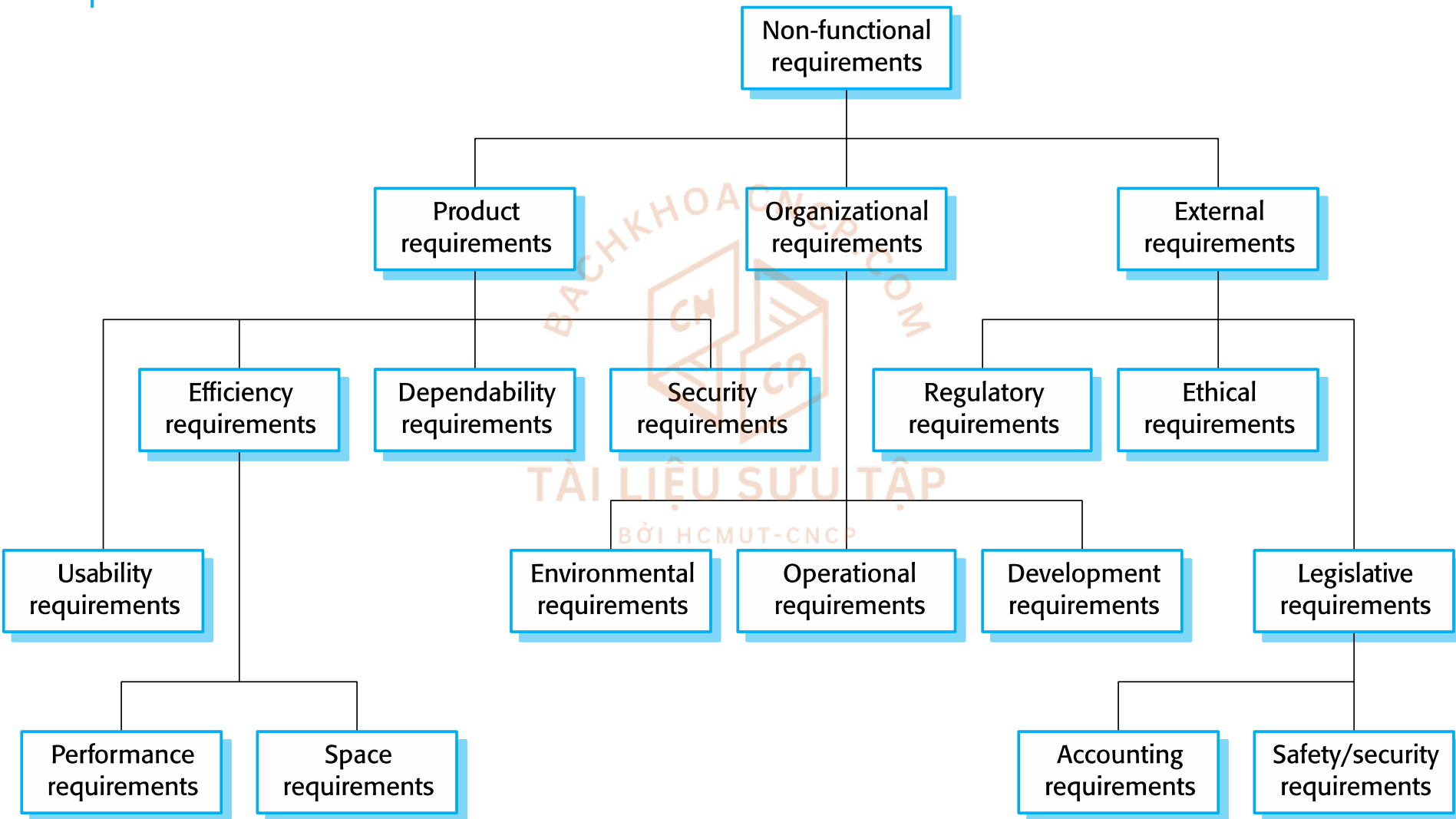
# NON-FUNCTIONAL REQUIREMENTS

✓ Define system properties and constraints
- Properties: reliability, response time and storage requirements.
- Constraints: I/O device capability, system representations, etc.

✓ Non-functional requirements may be more critical than functional requirements.
- If these are not met, the system may be useless.

# NON-FUNCTIONAL REQUIREMENTS IMPLEMENTATION

✓ Non-functional requirements may affect the overall architecture of a system
  - rather than the individual components.

✓ A single non-functional requirement
  - may generate a number of related functional requirements
  - and may also generate requirements that restrict existing requirements.

# TYPES OF NONFUNCTIONAL REQUIREMENT

# NON-FUNCTIONAL CLASSIFICATIONS

✓ Product requirements
- Requirements which specify that the delivered product must behave in a particular way e.g. execution speed, reliability, etc.

✓ Organisational requirements
- Requirements which are a consequence of organisational policies and procedures e.g. process standards used, implementation requirements, etc.

✓ External requirements
- Requirements which arise from factors which are external to the system and its development process e.g. interoperability requirements, legislative requirements, etc.

http://aita.gov.vn/Uploaded/file/CV%20HD%20huong%20dan%20YC%20phi%20chuc%20nang-130129.doc

# EXAMPLES OF NONFUNCTIONAL REQUIREMENTS IN THE MENTCARE SYSTEM

✓ ## Product requirement
- The Mentcare system shall be available to all clinics during normal working hours (Mon–Fri, 0830–17.30). Downtime within normal working hours shall not exceed five seconds in any one day.

✓ ## Organizational requirement
- Users of the Mentcare system shall authenticate themselves using their health authority identity card.

✓ ## External requirement
- The system shall implement patient privacy provisions as set out in HStan-03-2006-priv.

# GOALS VS. REQUIREMENTS

✓ Goal
  - A general intention of the user such as ease of use.

✓ Testable non-functional requirement
  - A statement using some measure that can be objectively tested.

# GOALS VS. REQUIREMENTS (CONT.)

**Goal:**
The system should be <u>easy to use</u> by medical staff.

**Non-functional requirement:**
Medical staff shall be <u>able to use</u> all the system functions <u>after four hours of training</u>.

# METRICS FOR SPECIFYING NONFUNCTIONAL REQUIREMENTS

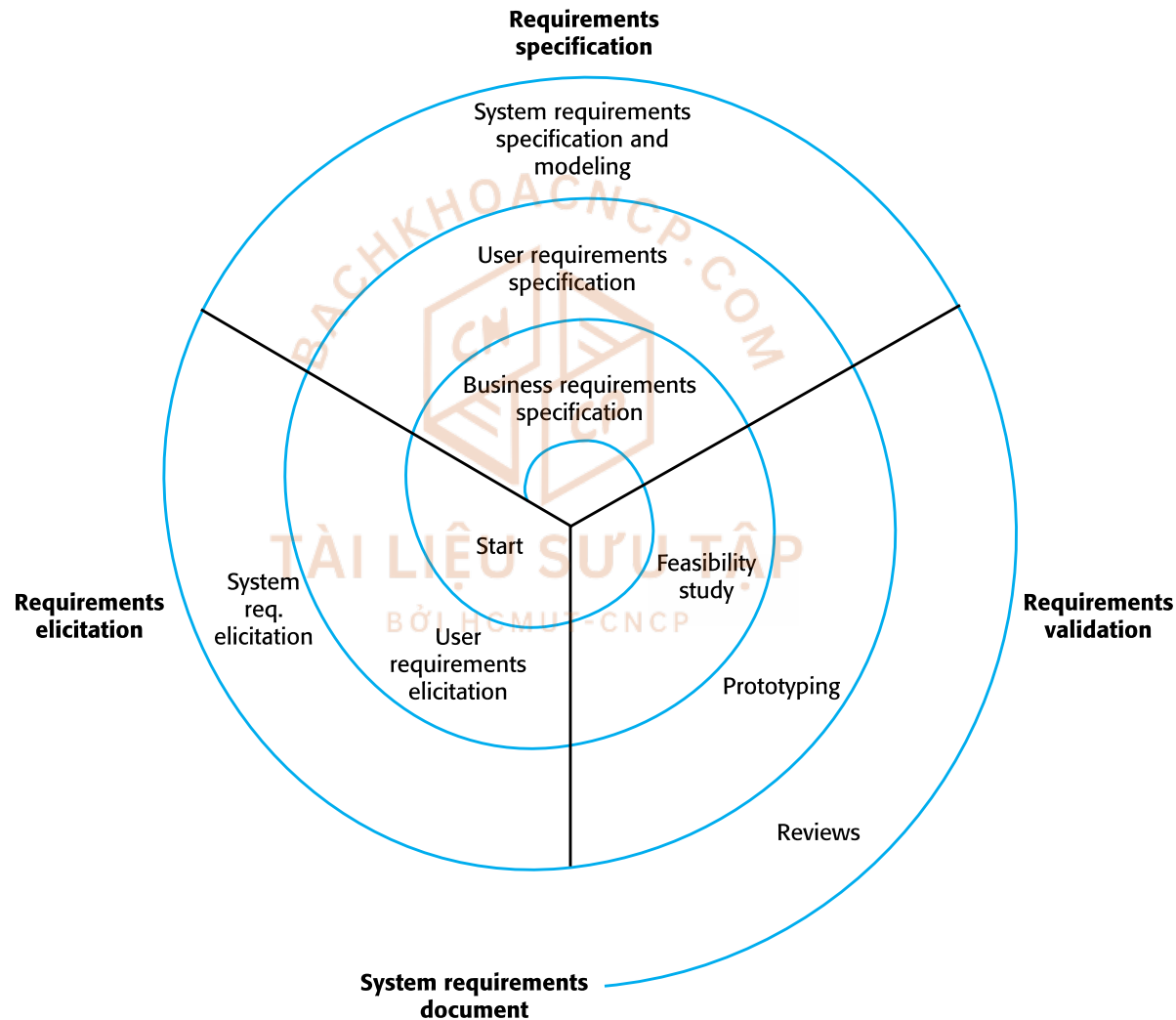| Property | Measure |
|---|---|
| Speed | Processed transactions/second<br>User/event response time<br>Screen refresh time |
| Size | Mbytes<br>Number of ROM chips |
| Ease of use | Training time<br>Number of help frames |
| Reliability | Mean time to failure<br>Probability of unavailability<br>Rate of failure occurrence<br>Availability |
| Robustness | Time to restart after failure<br>Percentage of events causing failure<br>Probability of data corruption on failure |
| Portability | Percentage of target dependent statements<br>Number of target systems |
| … | |

# REQUIREMENTS ENGINEERING PROCESSES

Week 5

# REQUIREMENTS ENGINEERING PROCESSES

✓ Processes to "generate" all requirements

✓ Generic activities common to all processes

- Requirements elicitation;
- Requirements analysis;
- Requirements validation;
- Requirements management.

✓ In practice, RE is an iterative activity

# A SPIRAL VIEW OF THE REQUIREMENTS ENGINEERING PROCESS

**Requirements specification**

System requirements specification and modeling

User requirements specification

Business requirements specification

Start

Feasibility study

**Requirements elicitation**

System req. elicitation

User requirements elicitation

Prototyping

**Requirements validation**

Reviews

**System requirements document**

# REQUIREMENT ELICITATION AND ANALYSIS

# REQUIREMENTS ELICITATION AND ANALYSIS

✓ ~ requirements elicitation or requirements discovery.

✓ Work with customers to find out:
  ▪ the application domain, the services and the operational constraints (system performance, hardware constraints, etc.).

✓ May involve
  ▪ end-users, managers, engineers involved in maintenance, domain experts, trade unions, etc. These are called *stakeholders*.

# PROBLEMS OF REQUIREMENTS ELICITATION

- ✓ Stakeholders don't know what they really want.

- ✓ Stakeholders express requirements in their own terms.

- ✓ Different stakeholders may have conflicting requirements.

- ✓ Organisational and political factors may influence the system requirements.

- ✓ The requirements change during the analysis process.
  - New stakeholders may emerge and the business environment change.

# THE REQUIREMENTS ELICITATION AND ANALYSIS PROCESS



Interacting with stakeholders to discover their requirements.

Requirements are documented and input into the next round

1. Requirements Discovery

4. Requirements Specification

2. Requirements Classification and Organization

Groups related requirements and organises them into coherent clusters

Prioritising requirements and resolving conflicts

3. Requirements Prioritization and Negotiation

# REQUIREMENTS DISCOVERY

✓ To gather information about the required and existing systems and distil the user and system requirements from this information.

✓ Main concerns:
- Stakeholders
- Discovery techniques/approaches/…

# DISCOVERY TECHNIQUE - INTERVIEWING

✓ Part of most RE processes.

✓ Types of interview
  ▪ Closed vs Open => mixed?

✓ Be effective
  ▪ Be open-minded, avoid pre-conceived ideas about the requirements and are willing to listen to stakeholders.
  ▪ Prompt the interviewee to get discussions going using a springboard question, a requirements proposal, or by working together on a prototype system.

# DISCOVERY TECHNIQUE - ETHNOGRAPHY

✓ Observational technique
- used to understand operational processes and help derive support requirements for these processes

✓ How
- A social scientist spends a considerable time observing and analysing how people actually work.
- People do not have to explain or articulate their work.
- Social and organisational factors of importance may be observed.

# STORIES AND SCENARIOS

- ✓ Scenarios and user stories are real-life examples of how a system can be used.

- ✓ Stories and scenarios are a description of how a system may be used for a particular task.

- ✓ Because they are based on a practical situation, stakeholders can relate to them and can comment on their situation with respect to the story.

# EX: PHOTO SHARING IN THE CLASSROOM (ILEARN)

Jack is a primary school teacher in Ullapool (a village in northern Scotland). He has decided that a class project should be focused around the fishing industry in the area, looking at the history, development and economic impact of fishing. As part of this, pupils are asked to gather and share reminiscences from relatives, use newspaper archives and collect old photographs related to fishing and fishing communities in the area. Pupils use an iLearn wiki to gather together fishing stories and SCRAN (a history resources site) to access newspaper archives and photographs. However, Jack also needs a photo sharing site as he wants pupils to take and comment on each others' photos and to upload scans of old photographs that they may have in their families.

Jack sends an email to a primary school teachers group, which he is a member of to see if anyone can recommend an appropriate system. Two teachers reply and both suggest that he uses KidsTakePics, a photo sharing site that allows teachers to check and moderate content. As KidsTakePics is not integrated with the iLearn authentication service, he sets up a teacher and a class account. He uses the iLearn setup service to add KidsTakePics to the services seen by the pupils in his class so that when they log in, they can immediately use the system to upload photos from their mobile devices and class computers.

# SCENARIOS

✓ A structured form of user story

✓ Scenarios should include
- A description of the starting situation;
- A description of the normal flow of events;
- A description of what can go wrong;
- Information about other concurrent activities;
- A description of the state when the scenario finishes.

# UPLOADING PHOTOS (ILEARN)

✓ **Initial assumption:** A user or a group of users have one or more digital photographs to be uploaded to the picture sharing site. These are saved on either a tablet or laptop computer. They have successfully logged on to KidsTakePics.

✓ **Normal:** The user chooses upload photos and they are prompted to select the photos to be uploaded on their computer and to select the project name under which the photos will be stored. They should also be given the option of inputting keywords that should be associated with each uploaded photo. Uploaded photos are named by creating a conjunction of the user name with the filename of the photo on the local computer.

✓ On completion of the upload, the system automatically sends an email to the project moderator asking them to check new content and generates an on-screen message to the user that this has been done.

# UPLOADING PHOTOS (ILEARN) (CONT.)

✓ **What can go wrong:**
- No moderator is associated with the selected project. An email is automatically generated to the school administrator asking them to nominate a project moderator. Users should be informed that there could be a delay in making their photos visible.

- Photos with the same name have already been uploaded by the same user. The user should be asked if they wish to re-upload the photos with the same name, rename the photos or cancel the upload. If they chose to re-upload the photos, the originals are overwritten. If they chose to rename the photos, a new name is automatically generated by adding a number to the existing file name.

✓ **Other activities:** The moderator may be logged on to the system and may approve photos as they are uploaded.

✓ **System state on completion:** User is logged on. The selected photos have been uploaded and assigned a status 'awaiting moderation'. Photos are visible to the moderator and to the user who uploaded them.

# REQUIREMENTS SPECIFICATION

# REQUIREMENTS SPECIFICATION

✓ The process of writing down the user and system requirements in a requirements document.

✓ Notes:

- User requirements have to be understandable by end-users and customers who do not have a technical background.

- System requirements are more detailed requirements and may include more technical information.

- The requirements may be part of a contract for the system development

# WAYS OF WRITING A SYSTEM REQUIREMENTS SPECIFICATION

| Notation | Description |
|---|---|
| Natural language | Sentences in natural language. Each sentence should express one requirement. |
| Structured natural language | Natural language statements on a standard form or template |
| Design description languages | Uses a language like a programming language, but with more abstract features |
| Graphical notations | Graphical models, supplemented by text annotations (best for functional requirements); UML use case and sequence diagrams are commonly used. |
| Mathematical specifications | Based on mathematical concepts such as finite-state machines or sets; Can reduce the ambiguity but hard to understand (and hard to check manually) |

# NATURAL LANGUAGE SPECIFICATION

✓ Used for writing requirements because it is expressive, intuitive and universal.
- The requirements can be understood by users and customers.

✓ Problems
- Lack of clarity: Precision is difficult without making the document difficult to read.
- Requirements confusion: Functional and non-functional requirements tend to be mixed-up.
- Requirements amalgamation: Several different requirements may be expressed together.

# EXAMPLE REQUIREMENTS FOR THE INSULIN PUMP SOFTWARE SYSTEM

Req 3.2. The system shall measure the blood sugar and deliver insulin, if required, every 10 minutes.

Req 3.6. The system shall run a self-test routine every minute with the conditions to be tested and the associated actions defined in Table 1.

# STRUCTURED SPECIFICATIONS

✓ Writing on a standard form or template:
- Name
- Inputs, outputs
- The information needed for the computation
- Action
- Pre and post conditions (if appropriate)
- The side effects (if any)

✓ This works well for some types of requirements e.g. requirements for embedded control system

| **Insulin Pump/Control Software/SRS/3.3.2** | |
|---|---|
| Function | Compute insulin dose: safe sugar level |
| Description | Computes the dose of insulin to be delivered when the current measured sugar level is in the safe zone between 3 and 7 units. |
| Inputs | Current sugar reading (r2); the previous two readings (r0 and r1). |
| Source | Current sugar reading from sensor. Other readings from memory. |
| Outputs | CompDose—the dose in insulin to be delivered |
| Destination | Main control loop. |
| Action | CompDose is zero if the sugar level is stable or falling or if the level is increasing but the rate of increase is decreasing. If the level is increasing and the rate of increase is increasing, then CompDose is computed by dividing the difference between the current sugar level and the previous level by 4 and rounding the result. If the result, is rounded to zero then CompDose is set to the minimum dose that can be delivered. |
| Requirements | Two previous readings so that the rate of change of sugar level can be computed |
| Pre-condition | The insulin reservoir contains at least the maximum allowed single dose of insulin. |
| Post-condition | r0 is replaced by r1 then r1 is replaced by r2. |
| Side effects | None |

# TABULAR SPECIFICATION

✓ Particularly useful when you have to define a number of possible alternative courses of action.

✓ Example:

| Condition | Action |
|---|---|
| Sugar level falling (r2 < r1) | CompDose = 0 |
| Sugar level stable (r2 = r1) | CompDose = 0 |
| Sugar level increasing and rate of increase decreasing ((r2 – r1) < (r1 – r0)) | CompDose = 0 |
| Sugar level increasing and rate of increase stable or increasing ((r2 – r1) ≥ (r1 – r0)) | CompDose = round ((r2 – r1)/4)<br>**If** rounded result = 0 **then**<br>          CompDose = MinimumDose |

# USE CASES

✓ Use-cases are a kind of scenario
   ▪ identify the actors in an interaction and which describe the interaction itself
   ▪ Included in the UML

✓ A set of use cases should describe all possible interactions with the system.

✓ UML sequence diagrams may be used to add detail to use-cases
   ▪ show the sequence of event processing in the system

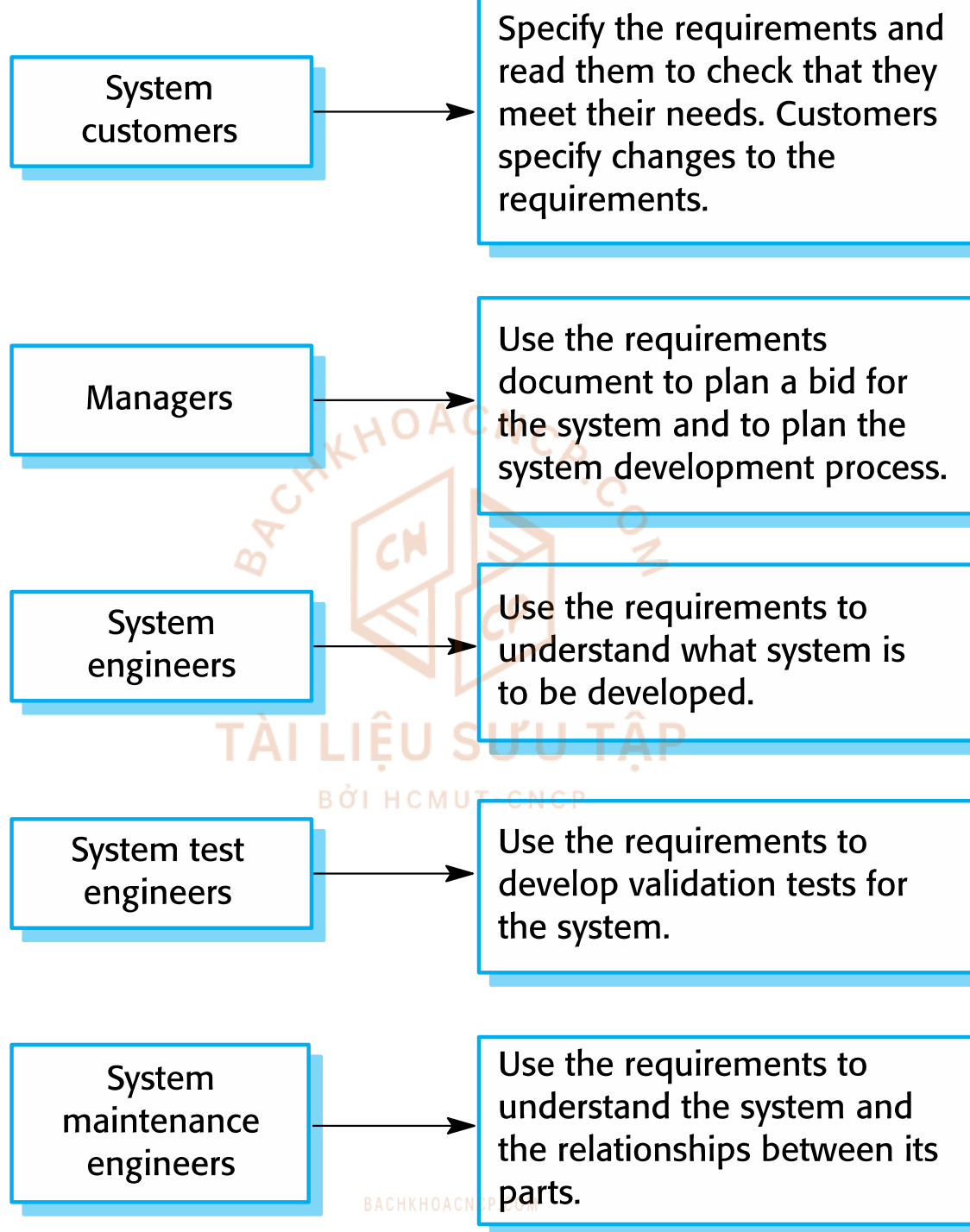# USE CASES FOR THE MHC-PMS
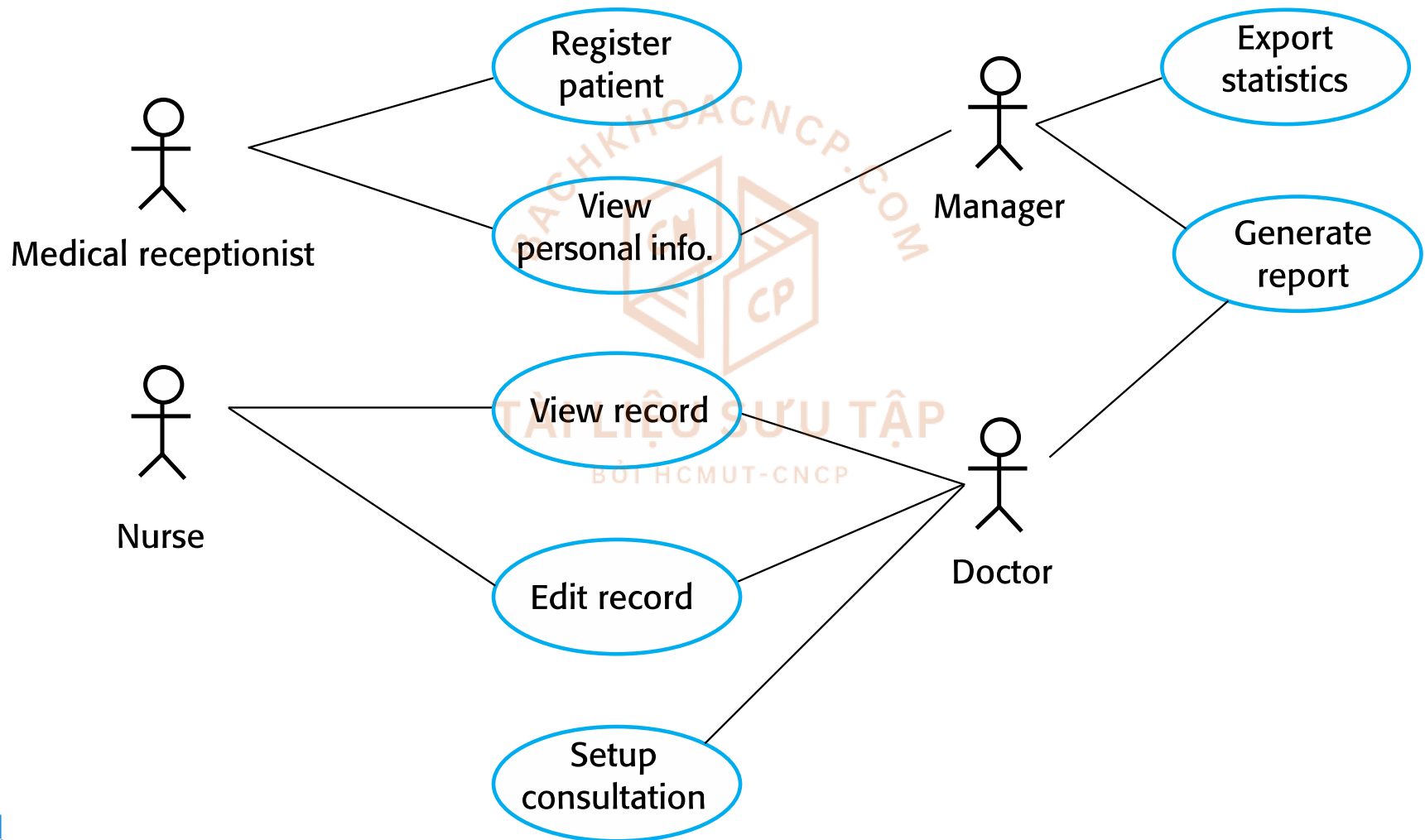
# THE SOFTWARE REQUIREMENTS DOCUMENT

✓ The software requirements document is the official statement of what is required of the system developers.

✓ Should include both a definition of user requirements and a specification of the system requirements.

✓ It is NOT a design document. As far as possible, it should set of WHAT the system should do rather than HOW it should do it.

# USERS OF A REQUIREMENTS DOCUMENT

| | |
|---|---|
| **System customers** | Specify the requirements and read them to check that they meet their needs. Customers specify changes to the requirements. |
| **Managers** | Use the requirements document to plan a bid for the system and to plan the system development process. |
| **System engineers** | Use the requirements to understand what system is to be developed. |
| **System test engineers** | Use the requirements to develop validation tests for the system. |
| **System maintenance engineers** | Use the requirements to understand the system and the relationships between its parts. |

# THE STRUCTURE OF A REQUIREMENTS DOCUMENT

| Chapter | Description |
|---|---|
| Preface | define the expected readership of the document and describe its version history |
| Introduction | describe the need for the system, briefly describe the system's functions, how the system fits into the overall business or strategic objectives |
| Glossary | define the technical terms used in the document |
| User requirements definition | describe the services provided for the user, the nonfunctional system requirements; may use natural language, diagrams, other notations that are understandable to customers; product and process standards that must be followed |
| System architecture | present a high-level overview of the anticipated system architecture, the distribution of functions across system modules |

# THE STRUCTURE OF A REQUIREMENTS DOCUMENT

| Chapter | Description |
|---------|-------------|
| System requirements specification | describe the functional and nonfunctional requirements in more detail |
| System models | might include graphical system models showing the relationships between the system components and the system and its environment |
| System evolution | describe the fundamental assumptions, any anticipated changes due to hardware evolution, changing user needs, … |
| Appendices | provide detailed, specific information that is related to the application being developed |
| Index | May include several indexes to the document, a normal alphabetic index; may be an index of diagrams, an index of functions,… |

# REQUIREMENT VALIDATION

# REQUIREMENTS VALIDATION

✓ Concerned with demonstrating that the requirements define the system that the customer really wants.

✓ Requirements error costs are high so validation is very important
- Fixing a requirements error after delivery may cost up to 100 times the cost of fixing an implementation error.

# REQUIREMENTS CHECKING

✓ Validity.
  - Does the system provide the functions which best support the customer's needs?

✓ Consistency.
  - Are there any requirements conflicts?

✓ Completeness.
  - Are all functions required by the customer included?

✓ Realism.
  - Can the requirements be implemented given available budget and technology

✓ Verifiability.
  - Can the requirements be checked?

# REQUIREMENTS VALIDATION TECHNIQUES

✓ Requirements reviews
  ▪ Systematic manual analysis of the requirements.

✓ Prototyping
  ▪ Using an executable model of the system to check requirements.

✓ Test-case generation
  ▪ Developing tests for requirements to check testability.

# REQUIREMENTS CHANGE

# CHANGING REQUIREMENTS

✓ The business and technical environment of the system always changes after installation.

✓ The people who pay for a system and the users of that system are rarely the same people.

✓ Large systems usually have a diverse user community, with many users having different requirements and priorities that may be conflicting or contradictory.

# REQUIREMENTS MANAGEMENT

✓ Requirements management is the process of managing changing requirements during the requirements engineering process and system development.

✓ New requirements emerge as a system is being developed and after it has gone into use.

✓ You need to keep track of individual requirements and maintain links between dependent requirements so that you can assess the impact of requirements changes. You need to establish a formal process for making change proposals and linking these to system requirements.

# REQUIREMENTS MANAGEMENT PLANNING

✓ **Establishes the level of requirements management detail that is required.**

✓ **Requirements management decisions:**

- ▪ **Requirements identification**
- ▪ **A change management process**
- ▪ **Traceability policies**
- ▪ **Tool support**

# REQUIREMENTS CHANGE MANAGEMENT

✓ Deciding if a requirements change should be accepted

- Problem analysis and change specification
- Change analysis and costing
- Change implementation

Identified problem → **Problem analysis and change specification** → **Change analysis and costing** → **Change implementation** → Revised requirements
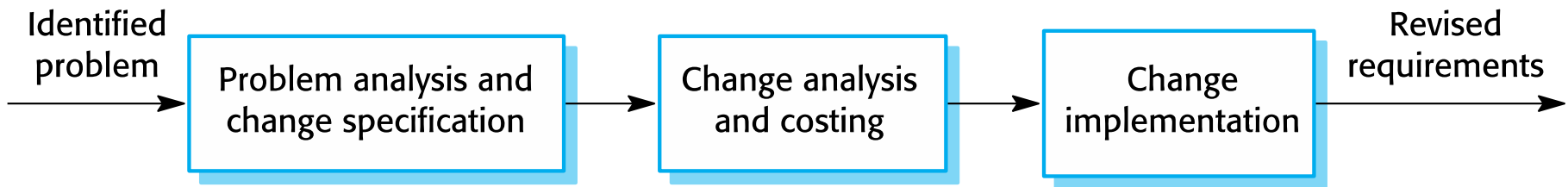
# SUMMARY

✓ Requirements: what the system should do and constraints on its operation and implementation.

✓ Functional requirements = the services

✓ Non-functional requirements = constraints (development & use)
   ▪ apply to the system as a whole.

✓ The software requirements document (i.e. SRS) is an agreed statement of the system requirements.

✓ The RE process is an iterative process
   ▪ requirements elicitation, specification and validation.

# SUMMARY (CONT.)

- ✓ **Requirements elicitation and analysis = iterative process**
  - ▪ requirements discovery, classification and organization, negotiation and requirements documentation.

- ✓ **Techniques for requirements elicitation**
  - ▪ interviews, scenarios, use-cases and ethnography, etc.

- ✓ **Requirements validation = checking the requirements**
  - ▪ for validity, consistency, completeness, realism and verifiability.

- ✓ **Business, organizational and technical changes inevitably**
  - ▪ => changes to the requirements for a software system.

- ✓ **Requirements management = managing and controlling the requirement changes.**